# Hybrid information flow analysis against web tracking (invited talk)

Thomas Jensen

Joint work with:

Frédéric Besson

Nataliia Bielova

# Hybrid information flow analysis against web tracking

Thomas Jensen, Inria

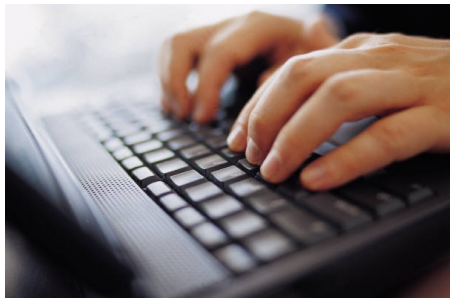
Keynote, CRISIS'2017

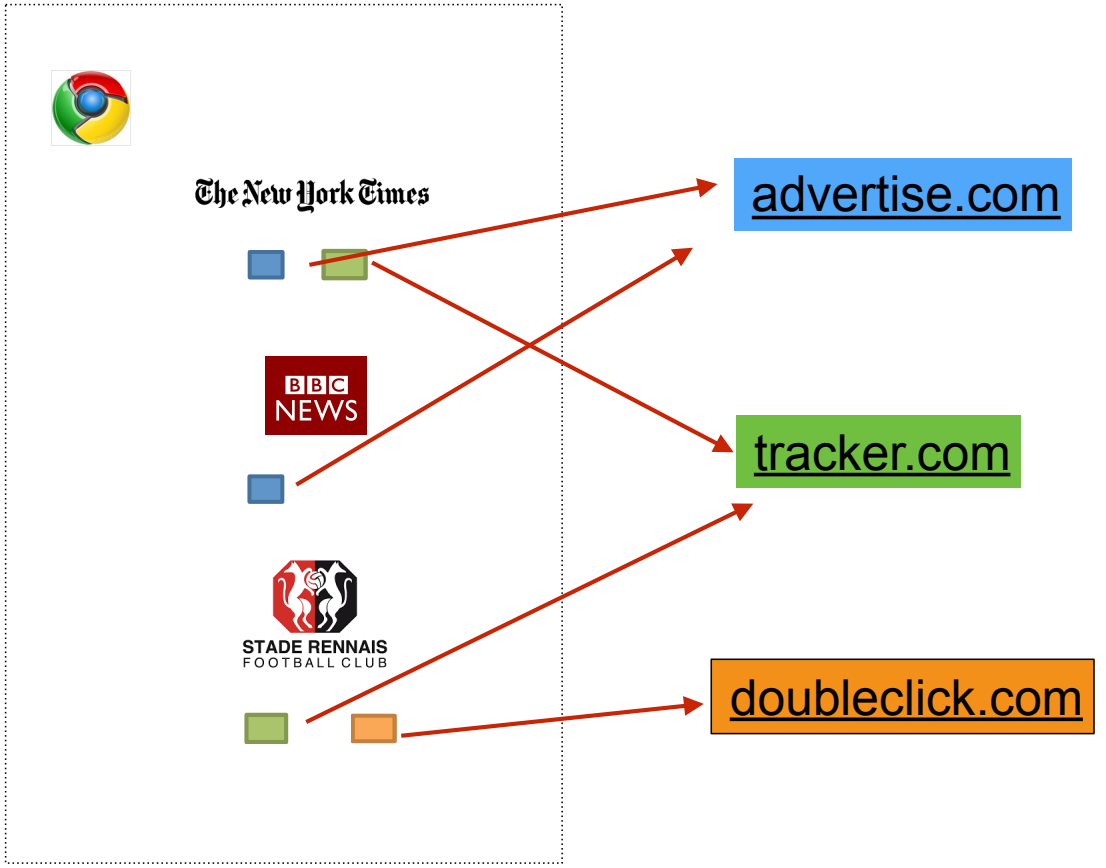
# Overview

1. Web tracking.
2. Knowledge-based information flow analysis.
3. Browser randomisation.

Thanks to EU ePrivacy Directive

# Web tracking






Bigger browsing profiles

= increased value for trackers
= reduced privacy for users

(Hypothetical tracking relationships only.)

# Browser extensions to the rescue?

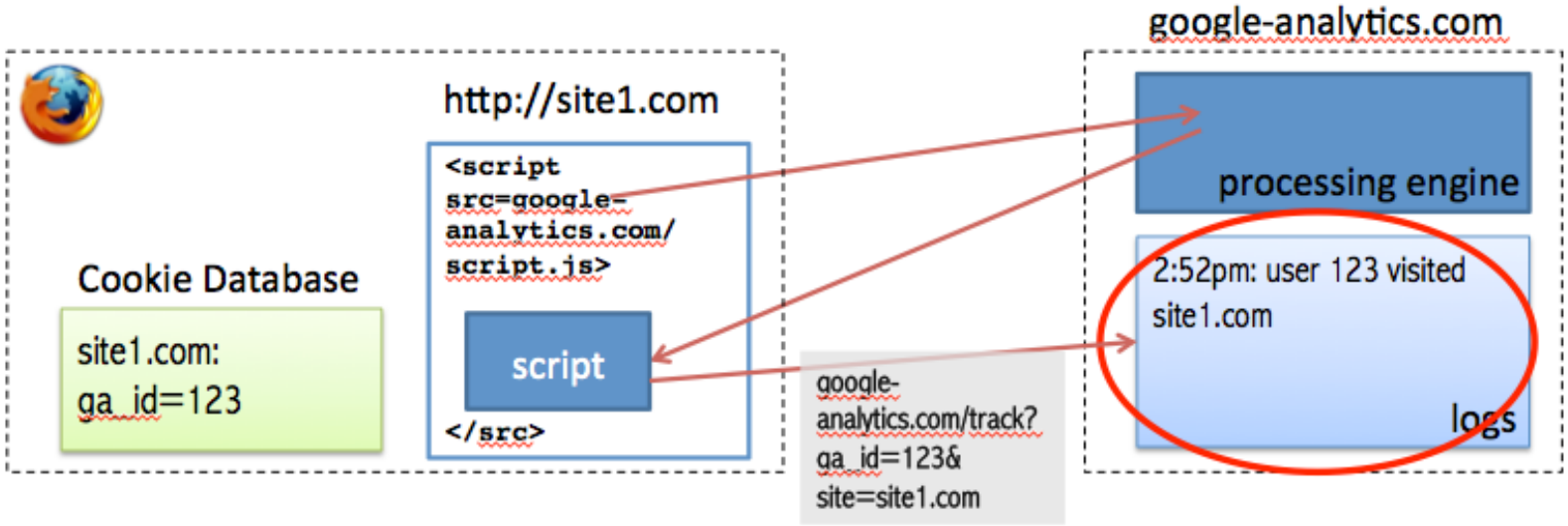

AdBlockPlus: blocks scripts/requests only from known advertisement companies

Ghostery: blocks scripts/requests only from known tracking companies

- No protection from tracking by other companies
- No protection from tracking by the main (first-party) web site

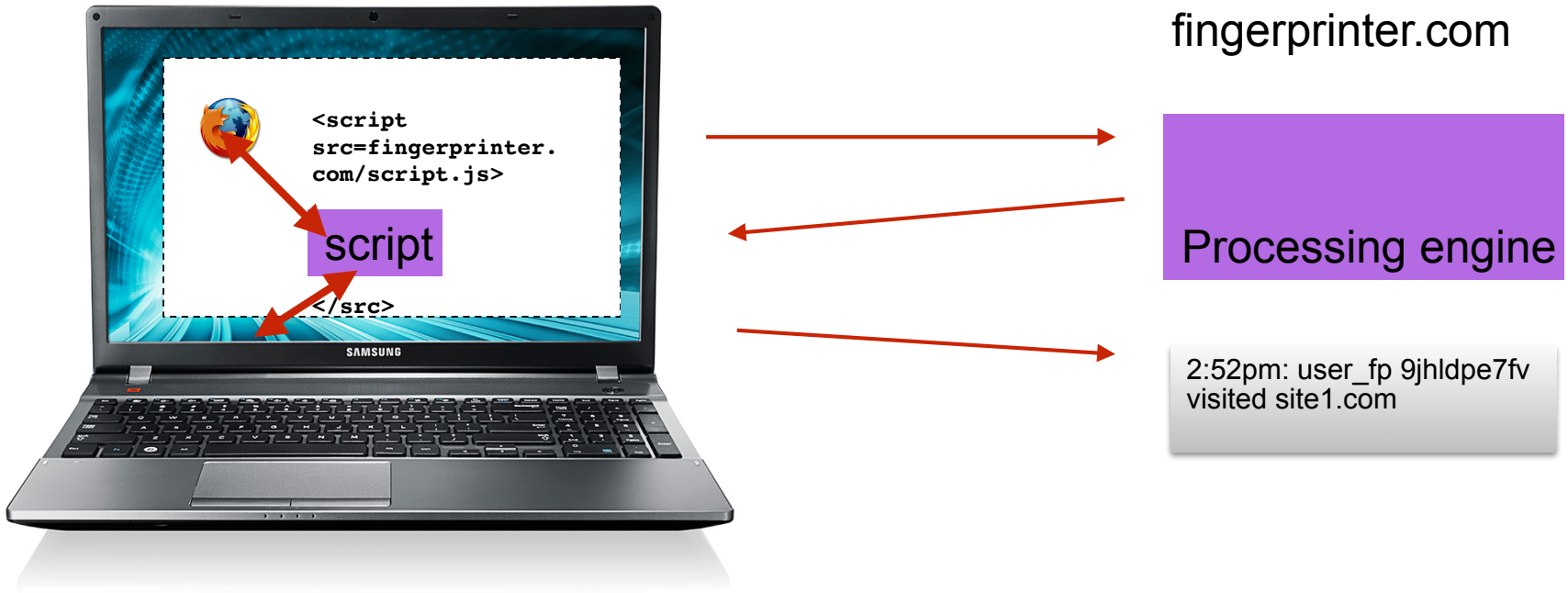
# Tracking by storing identity

Cookies are used to track repeated visits to a site.



# Tracking by finger-printing

Browser and operating system properties are used to track repeated visits to a site.



# Panopticlick: an early study on finger-printing


- A study from 2010 showed that 83,6 % of browser fingerprints were unique among the 500 000 browsers tested.
- Fingerprints: HTTP headers, browser and OS features,: language, plug-ins, fonts, …

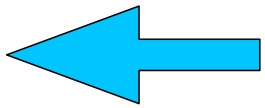
# Protection mechanisms

Stateful tracking: well-known and being addressed at different levels:

- Third-party cookie blocking
- EU e-Privacy directive
- Non-interference analysis for JavaScript

Stateless tracking: less clear

- IP address tracking
- Web browser finger-printing



# 2. Knowledge-based information flow analysis

# Use information flow to spot web tracking

Scripts can access information about configurations

Goal: Estimate the information that a script learns about configurations.

But some information leakage is reasonable!

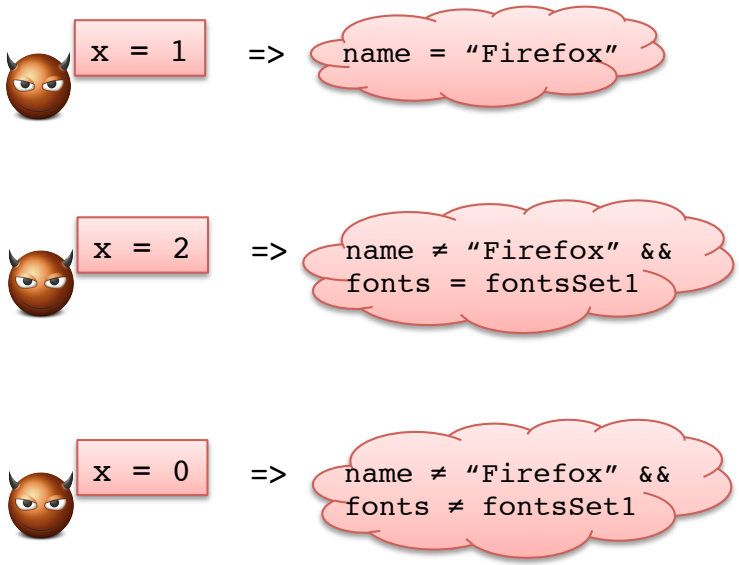
Scripts may need information about browser and execution environment for optimal user experience.

# Information flow control

| Non-interference | Attacker knowledge | Quantitative information flow |
|---|---|---|
| Secret input does not flow into public output<br><br>Yes/no ? | What information about the secret flows to output in an execution.<br><br>Secret > 42 | How much information about secrets is leaked?<br><br>Leaks 17 bits |

# Attacker knowledge

```
var x = 0;
if (name == “Firefox”) {
    x = 1;
}
else {
    if (fonts == fontsSet1) {
        x = 2;
    }
}
output x;
```




Depending on the browser, different executions leak different amount of information.

# Non-interference

A program is termination-insensitive non-interferent if it produces the same output for all low-equivalent input.

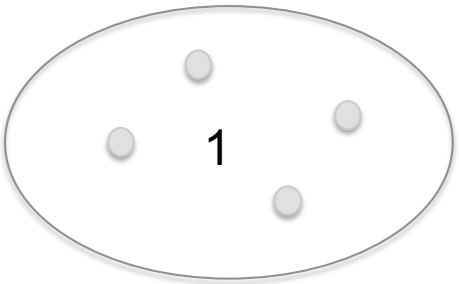

Example with two binary secrets h1, h2 ∈ {0,1}

# Secure and insecure executions

The following program is not secure - but it has secure executions.

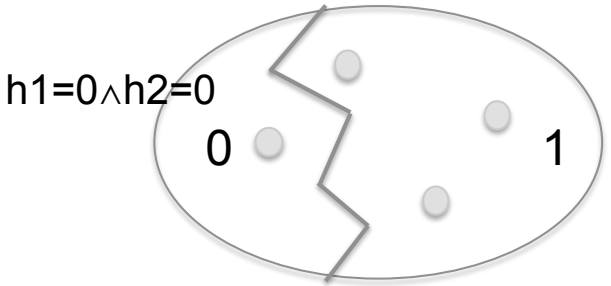
```
if h1 = 1 then x = 1
else skip;
if h2 = 1 then low = 1
else low = x;
output low
```

Initial values: [low = 1, x = 1]




Secure

[low = 1, x = 0]




Leaks information about secret

[Zdancewic'02, Austin & Flanagan'10]

# Monitoring information flow

## Dynamic information flow control

- Label values with security labels.
- Propagate labels during execution.
- Halt execution before outputting secret values.

Non-interference needs "no-sensitive- upgrade" principle:
no assignment to public variables under secret control

[secret = 1]

```
if secret = 1 then
  output = 1
else output = public;
return output
```

[Le Guernic'07, Russo & Sabelfeld'10,…]

# Hybrid information flow analysis

Combine dynamic IF control with a static analysis

- statically analyse the non-executed branches
- admits more executions but limited by the precision of the static analysis

[secret = 1, public = 1]

```
if secret = 1 then
  output = 1
else output = public;
return output
```

Static analysis says
`public : [1]`

# Hybrid monitor of attacker knowledge

Define a hybrid monitor that tracks the knowledge of an attacker.

Use knowledge to

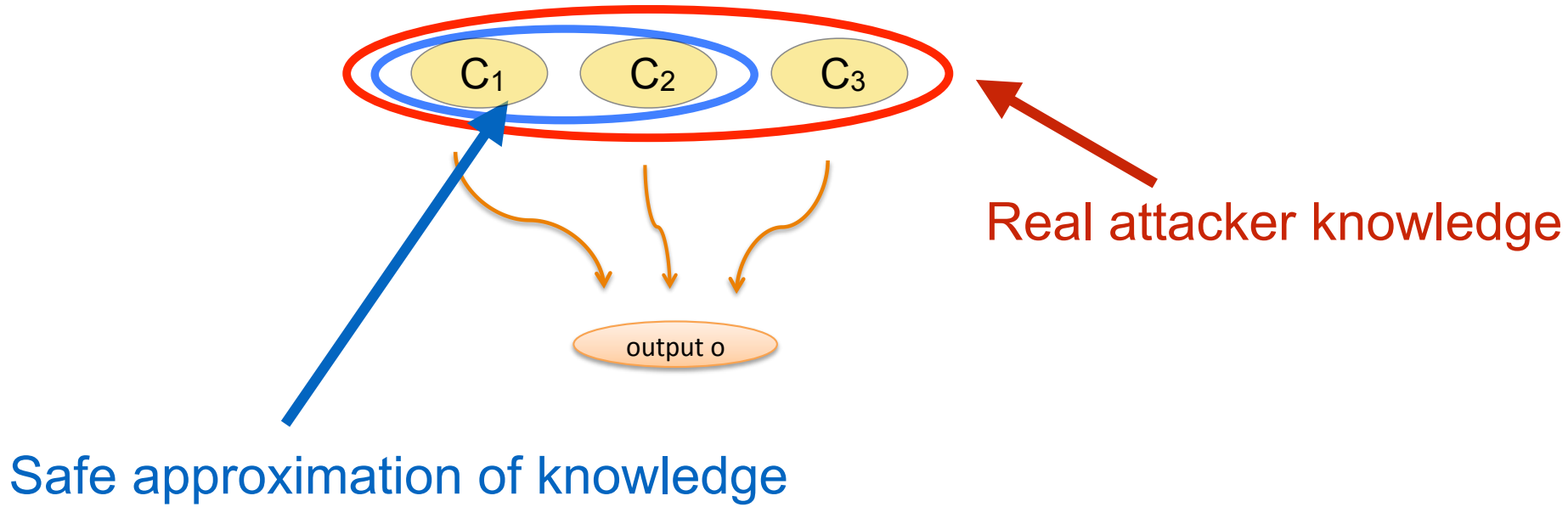
- enforce non-interference,
- accept more secure executions.

Solution:

- A domain for representing attacker knowledge.
- Monitor formalised as an operational semantics.

# Attacker knowledge

Knowledge: What does an attacker learn about the input state by seeing a particular output.



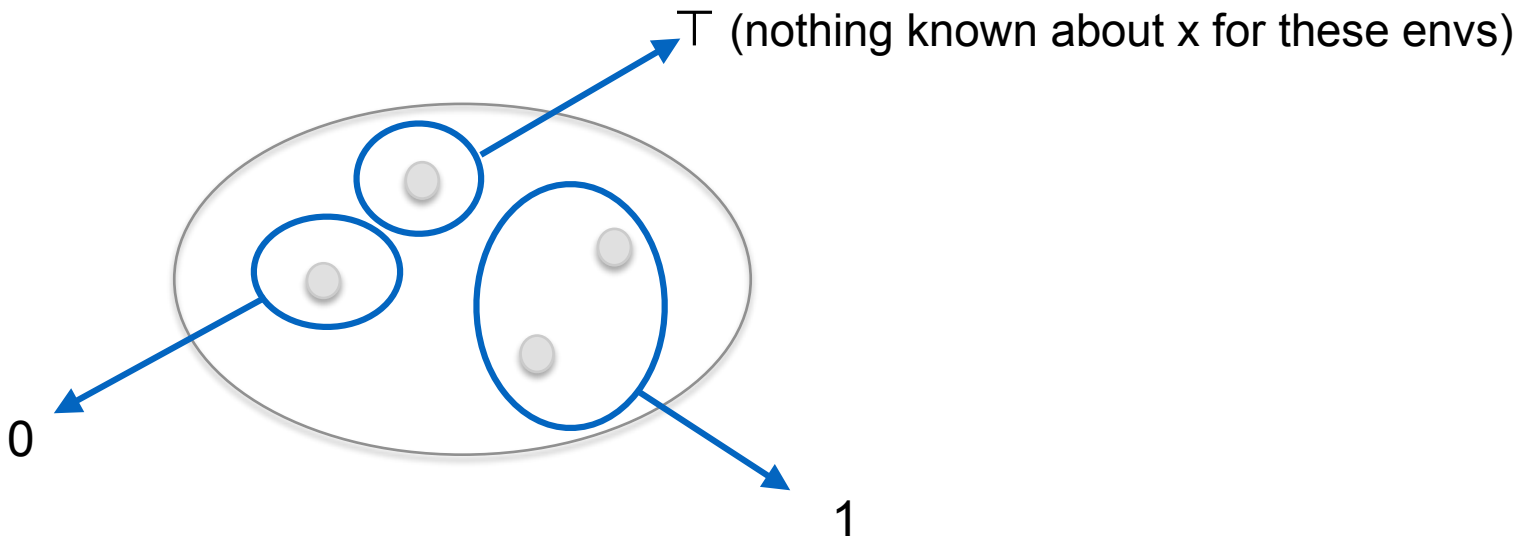
# Domain of knowledge

$$\kappa: \mathrm{Var} \rightarrow \mathbf{K}$$

$$\mathbf{K} \triangleq \mathrm{Env} \rightarrow \mathrm{Value} \cup \{\top, \bot\}$$

The knowledge in a variable $\kappa$(x):

which initial environments leads to a particular value of x?



# The hybrid monitor

Defined as an operational reduction semantics:

$$(P, \rho, \kappa) \Downarrow (\rho', \kappa')$$

- $\rho$, $\rho'$ : Env $\cup$ {$\cdot$} ,
- $\kappa$, $\kappa'$ : Var $\rightarrow$ K  labeling of variables with knowledge,
- Env : Var $\rightarrow$ Val  for dynamic analysis,
- $\cdot$ "empty" environment for static analysis.

# Semantic rules

Rules come in pairs: one dynamic and one static

**Dynamic analysis computes real values and knowledge**

**Static analysis called on-the-fly and propagates only knowledge**

$$\textsc{ifDyn}\ \frac{C[\![e]\!]_\rho = \alpha \qquad (c_\alpha, (\rho, \kappa)) \Downarrow (\rho', \kappa_\alpha) \qquad (c_{\bar{\alpha}}, (\cdot, \kappa)) \Downarrow (\cdot, \kappa_{\bar{\alpha}})}{(\text{if } e \text{ then } c_{tt} \text{ else } c_{ff}, (\rho, \kappa)) \Downarrow (\rho', \mathbb{IF}((\!|e|\!)_\kappa, \kappa_{tt}, \kappa_{ff}))}$$

$$\textsc{ifStat}\ \frac{(c_{tt}, (\cdot, \kappa)) \Downarrow (\cdot, \kappa_{tt}) \qquad (c_{ff}, (\cdot, \kappa)) \Downarrow (\cdot, \kappa_{ff})}{(\text{if } e \text{ then } c_{tt} \text{ else } c_{ff}, (\cdot, \kappa)) \Downarrow (\cdot, \mathbb{IF}((\!|e|\!)_\kappa, \kappa_{tt}, \kappa_{ff}))}$$

Operator combining knowledge obtained in branches with knowledge from the condition

# Enforcing non-interference

Before any output, the monitor checks non-interference:

- if non-interference then output is not modified.

$$\textsc{outNI}\ \frac{(c, init(\rho)) \Downarrow (\rho', \kappa) \quad \rho'(x) = v \quad NI(\rho, \kappa(x), v)}{(c; \mathsf{output}\ x, \rho) \Downarrow v}$$

The non-interference check is defined as:

$$NI(\rho, K, v) \triangleq [\rho]_{\mathsf{L}} \subseteq K^{-1}(v) \cup K^{-1}(\bot).$$

set of low-equivalent env's

# Example analysis

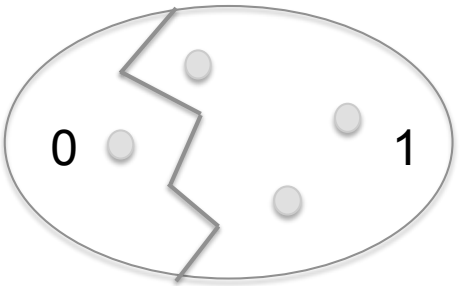
[h1 = 0 , h2 = 1]

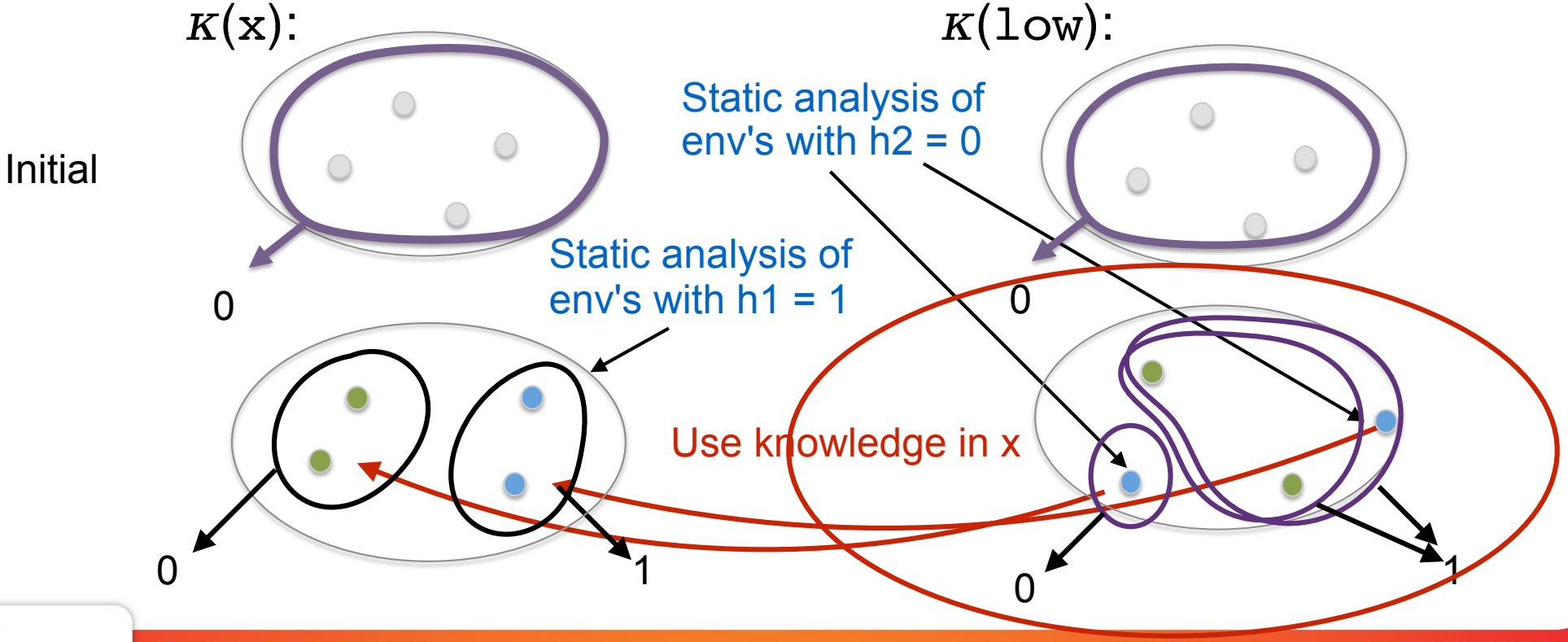
Static analysis

```
if h1 = 1 then x = 1
 else skip;
if h2 = 1 then low = 1
 else low = x;
output low
```

Knowledge

[low = 0 , x = 0]



# Stock-taking

Knowledge-based information flow analysis

- computes a correct approximation of attacker knowledge,
- enforces non-interference,
- more permissive than existing hybrid monitors,
- can be combined with other monitors to enhance precision.

# 3. Browser randomisation against web tracking

# How to enforce privacy?

The hybrid monitor monitor evaluates how much a tracker learns about an individual user.

How to react when a script tracks too much?

```
p(name) =
        Firefox   0.45
        Chrome 0.45
        Opera    0.10
```

- With 10 users, our Opera user will be uniquely identified.
- Halting the program or suppress output might still make you identifiable.
- Need collaboration between users.

# A solution

- Users switch between configurations
- How many configurations are needed to guarantee privacy?
- Usability issue: users want to switch as little as possible

Challenge: find a distribution on configurations for all users such that

- Privacy is guaranteed
- Usability is maximised

# Privacy: probability of guessing

```
if (name == "Opera") x = A;
else x = B;
output x;
```

A priori

| p(name) = | |
|---|---|
| Firefox | 0.45 |
| Chrome | 0.45 |
| Opera | 0.10 |

x = A

A posteriori

| p(name) = | |
|---|---|
| Firefox | 0 |
| Chrome | 0 |
| Opera | 1 |

Probability $Pr^G$ : guessing the secret in case of worst observation.

Stronger that definition based on Bayesian risk:

$$p(A)\ \max_i p(i|A) + p(B)\ \max_i p(i|B) = 0.55$$

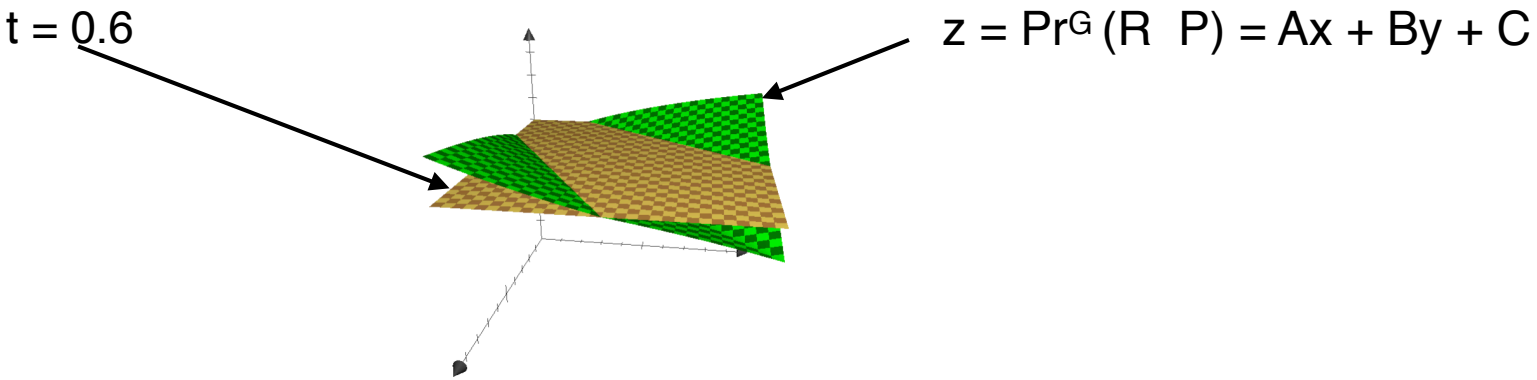
# Achieving privacy

Find a randomisation R of user state such that when composed with tracking script P

$$Pr^G ( R ; P ) < \text{privacy threshold.}$$

| R | "Firefox" | "Opera" |
|---|---|---|
| $id_1$ | x | 1-x |
| $id_2$ | 1-y | y |

| P | o1 | o2 | o3 |
|---|---|---|---|
| "Firefox" | 1/2 | 1/3 | 1/6 |
| "Opera" | 1/6 | 1/2 | 1/3 |



# Adding usability

Users don't want to switch configuration:

- choose randomisation so that they get their original configuration as much as possible
- linear constraint on the randomisation matrix.
- problem reduces to linear programming.

| R | "Firefox" | "Opera" |
|---|---|---|
| $id_1$ | x | 1-x |
| $id_2$ | 1-y | y |

Usability

$$\max (x + y) \text{ s.t.}$$
$$0 \leq x \leq 1$$
$$0 \leq y \leq 1$$
$$Ax + By + C \leq t$$

Privacy

# Wrapping up

Web tracking is done by different techniques

- cookies, other browser storage, fingerprinting

Information flow analysis against fingerprinting

- Hybrid information flow monitoring with static analysis
- Computes tracker knowledge.

Enforcing browser anonymity

- Randomisation of configurations
- Combining privacy and usability

# Based on a true story…

- F. Besson, N. Bielova, T. Jensen: Hybrid Information Flow monitoring against web tracking, CSF 2013.
- F. Besson, N. Bielova, T. Jensen: Browser randomization against fingerprinting: A Quantitative information flow approach, NordSec 2014.
- F. Besson, N. Bielova, T. Jensen: Hybrid monitoring of attacker knowledge, CSF 2016.